SOFTWARE QUALITY ASSURANCE AUDITS

GUIDEBOOK

NOVEMBER 1990

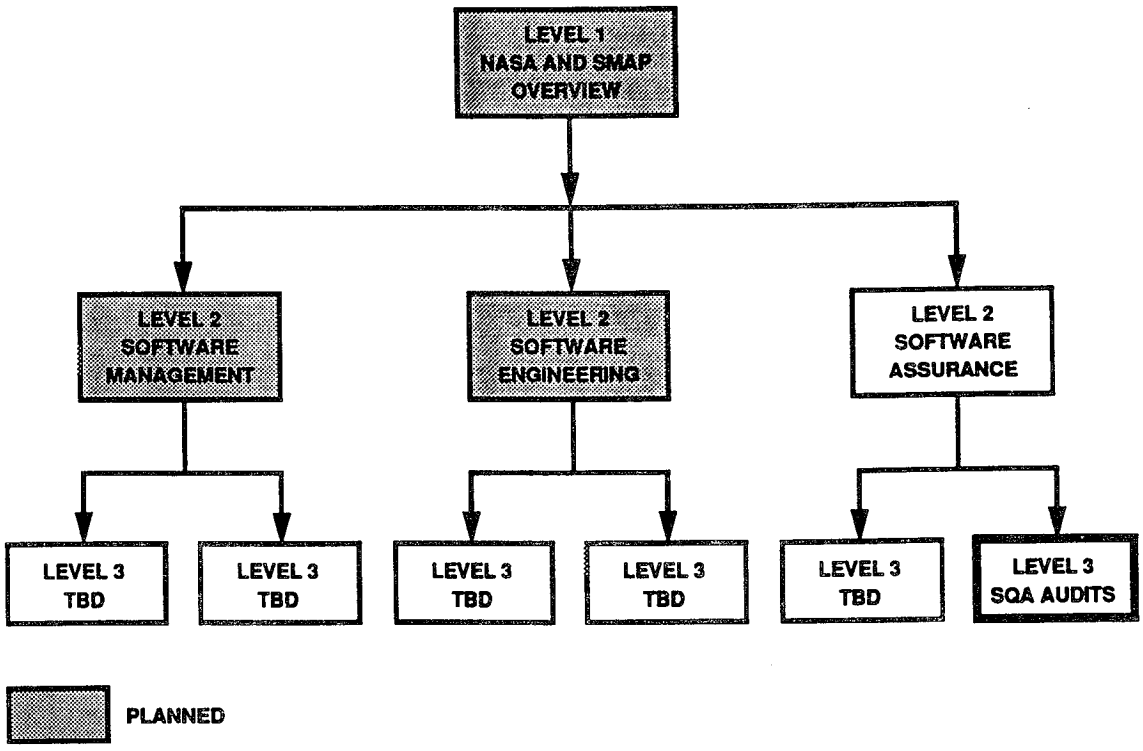
Office of Safety and Mission Quality
Software Management and Assurance Program (SMAP)
Guidebook Working Group

PREFACE

The growth in cost and importance of software has made it necessary for NASA to establish software standards and guidance for use in the development and acquisition of software. The Software Management and Assurance Program (SMAP), established in the Office of Safety and Mission Quality of NASA Headquarters, focuses the NASA activities in defining standards for software management, engineering, and assurance. One of the products of the SMAP is a series of guidebooks that defines a NASA concept of the processes that are used to manage, engineer, and assure software.

There are three levels of SMAP software guidebooks. Level 1 is reserved for a high level guidebook that will describe the NASA view of software and the SMAP. There will be three Level 2 guidebooks that will provide an overall picture of the concepts and practices of NASA in software management, assurance, and engineering. Level 3 guidebooks will focus on specific activities that fall within each of those three software disciplines, and provide more detailed information for the manager and/or practitioner.

This is the Level 3 Software Quality Assurance Audits Guidebook that describes software quality assurance audits in a way that is compatible with practices at NASA Centers. For a more generalized view of how software quality assurance audits relate to Software Assurance, refer to the Level 2 Software Assurance Guidebook, document number SMAP-GB-A201.



SOFTWARE QUALITY ASSURANCE AUDITS
GUIDEBOOK

Approvals

Lawrence E. Hyatt Nov 27, 1990
Lawrence E. Hyatt Date
Chairman, Software Assurance
Guidebook Working Group

Donald W. Sova 11/27/90
Donald W. Sova Date
Manager, SMAP

Carl Schneider 12/10/90
Carl Schneider Date
Director, RM&QA Division

TABLE OF CONTENTS

TABLE OF CONTENTS (Cont.)

I. GENERAL

The NASA Software Assurance Guidebook classifies the software quality assurance (SQA) audit as a fundamental quality assurance technique. It is the intent of this guidebook to further define audits, describe the audit process, and provide a sample checklist that can be tailored for use in an audit. The guidebook is written for quality assurance practitioners who will perform audits, software developers who will be audited, and for software project managers and acquirers who have to decide the extent of auditing to be done.

In this guidebook, the term "audit" specifically refers to an SQA technique that is used to examine the conformance of a development process to procedures and the conformance of products to standards. An SQA audit also can examine the conformance of the actual status of the development activity to the reported status. The term "audit" is used to describe a number of additional software activities; however, due to their different purpose and focus, they are not addressed in this guidebook. For example, the Functional Configuration Audit (FCA) and Physical Configuration Audit (PCA) are configuration management (CM) activities. Quality (Engineering) Audits and Safety Audits are technical activities that evaluate a software product against Quality Engineering and Safety requirements. These types of audits are not covered in this guidebook.

II. CONCEPTS AND DEFINITIONS

An SQA audit is an activity that is performed to determine the adherence to, and adequacy of, a project's established software development standards and procedures and the effectiveness of their implementation. As used in this guidebook, the main objective of an SQA audit is to determine the adherence to established standards and procedures; checking their adequacy or effectiveness is a secondary objective that usually is not requested of an auditor.

In the NASA Software Assurance Guidebook, standards are defined as "the established criteria to which software products are compared." Software standards include documentation standards, design standards, and coding standards. In that guidebook, procedures are defined as the "established criteria to which the development and control processes are compared." Procedures, then, are the step-by-step directions that are to be followed to accomplish some development or control process; for example, CM or nonconformance reporting and corrective action (NRCA). In other words, standards and procedures are requirements for software management, engineering, and assurance; SQA audits verify their existence and assess a project's compliance with them.

SQA audits also can compare the actual status of a product with reported status. Status auditing is most effective if there are objective and consistent criteria for evaluating the level of product completeness. For example, Unit Development Folders (UDFs) have a cover sheet for recording the progress of a unit through its development stages; the folder contains the actual product. If a project uses UDFs, then an audit can compare the actual product to the cover sheet and to the progress report.

The actual processes and products examined by an audit will vary depending on the objective of the audit. The objective of the audit can vary, and is determined by the organization that called for the audit. A general audit provides a comprehensive overview, while a limited audit might be an examination of certain procedures, such as CM, or a check on a certain requirement, such as "Are coding standards being followed?"

An audit may be described as internal or external, depending on the organization of origin of the auditor(s). An internal audit is an audit conducted by the SQA staff of the software developer. Internal audits are intended to be preventative in nature; to detect problems before they become major.

An external audit is one performed by an independent auditor who is outside of the developing organization. External audits are most often requested by the acquiring organization, as a means of obtaining an independent opinion about the work in progress. External audits tend to be more comprehensive in nature than internal

audits, and usually encompass a broad area of the development activity. Such audits usually are requested because the acquirer is uncertain of the effectiveness of the internal program or because of lack of information and fears about the quality of performance on the part of the developer. An advantage of an external audit is that the auditor may be more objective about a project than an internal auditor; however, an external auditor must spend more time learning about the project and its development process.

III. CONDUCTING AN SQA AUDIT

An SQA audit has four phases: planning and preparation, the site visit, reporting, and follow-up. During the planning and preparation phase, the auditor gains an understanding of the project. Based on the scope of the audit, the auditor determines the specific questions that need to be answered, as well as the persons to be interviewed and the records and products to be examined to answer the questions. The interviews are conducted, and records and products are examined during the site visit. The reporting phase consists of the exit debriefing of the audited project, the preparation of a written report on the audit, and clarifying issues and providing related information as needed. Follow-up is done by the project, as the problems and deficiencies found in the audit are remedied. Follow-up may include reauditing to assess the adequacy of the remedies.

The activities conducted during the phases vary depending on the life cycle phase of the project being audited and the scope of the audit. The activities also vary depending on whether the audit is external or internal; an external audit requires more extensive preparation and should examine a more comprehensive sample of material than an internal audit.

Each of the four phases of an audit is described in the following sections. The activities of each phase are described as if a general, external audit is to be done since this results in the greatest detail. Some of the activities may be superfluous to an internal SQA audit and may be omitted.

A. Audit Planning and Preparation

A general SQA audit should be planned carefully to examine all of the software engineering, management, and assurance processes and all of their products. Software management processes include status reporting and CM. Engineering processes include analysis, design, and code. Assurance processes include verification and validation (V&V) and NRCA. Products include documents and code. If the scope of the audit is more limited, then planning will be within the defined limits. A limited audit might examine only one of the processes or a limited set of products. Activities during the planning and preparation phase are similar for all audits, but preparation for a limited audit is focused on the identified process or product.

As a first step, the auditor should understand the objective of the software development project and what products are to be produced. The auditor needs to know what the contract requires in the way of deliverable software and documentation, and what, if any, requirements exist for management, engineering, and assurance practices. One source of this information may be the statement of work and other contract documents. Once it is clear what is being developed and what the contract requires, the auditor should review management documentation,

such as the software management, development, and assurance plans to understand the processes that will be used to develop and control the products. Then the developer's standards and procedures manual should be reviewed to determine the quality standards and the detailed procedures planned to be applied to the software and the development process. From this background information, the auditor should be able to understand the developer's software development process.

The auditor also should review some recent status reports from the developer. These reports will furnish information on the stage of completeness of products and may contain information as to problem areas.

After background familiarization and a look at project status, the auditor should define the areas that will require the most careful and detailed attention, i.e., the processes or products that seem to be in some difficulty or whose status is in doubt. These areas may be identified by the status reports, discussions with the acquirer of the software (if it is the acquirer who has requested the audit), review of nonconformance reports, and the results of previous audits.

Once the auditor understands the project and has identified the areas of concentration, he/she should develop a checklist. A checklist is a list of items to be examined and questions to be asked. Each checklist should be tailored for the specific project being audited and its life cycle phase and should reflect the scope of the audit. A more comprehensive and less detailed checklist is required for a general audit; a limited audit requires a checklist that is more detailed in specific areas. Guidance on preparing a checklist is given in Chapter VI. A checklist is intended to provide the auditor with a "road map" during the site visit. It must be complete, so that the auditor can know that sufficient information has been gathered if all of the checklist items are completed. The checklist questions help define the individuals with whom the auditor wishes an interview and the types of records that the auditor will examine.

The auditor should schedule the site visit to the project through its assurance staff or other suitable contact after the preparation is done and the checklist prepared. During this contact with the project, the auditor should specify the intent of the audit, the records to be examined, and which people the auditor wishes to interview. People to be interviewed will include managers, selected developers, CM staff, assurance staff, and testers. Copies of the checklist may be furnished to increase the project's understanding. The project should be prepared to provide the auditor with a convenient working area that includes normal office facilities, access to all products and records, and interviews with the identified individuals.

B. The Site Visit

The purpose of the audit site visit is to collect the data necessary to assess that the required products are being produced, the degree to which they conform to applicable standards, how well procedures are being followed, and that the reported status corresponds to the actual status. The audit is intended to uncover any significant deviation from standards, procedures, or reported status so that corrective action can be taken. The auditor uses two basic techniques: interviews with project staff and examination of documentation and records.

The site visit should begin with an entrance briefing, involving the auditor and key project staff. During this briefing, the auditor should describe the focus of the audit, and identify the interviews to be conducted and the records to be examined. The entrance briefing may also be used by the project to brief the auditor on its processes, key staff members, and current status. Time for questions and answers should be included. The auditor also should assure the project that an exit interview will be held where the auditor will present preliminary findings to the project and the project may provide any additional information to the auditor. This preliminary exchange of information can significantly help to allay the fears of the project and to smooth the course of the site visit.

After the entrance briefing, the auditor should proceed with the gathering of information. It is useful to begin the information gathering process with interviews, during which the auditor tries to understand the realities behind the documented plans and procedures. The auditor should learn which individuals carry out a procedure, approve a change or fix, keep project records, etc. Each individual should be asked to describe his/her perceptions of and interactions with the process. The auditor should take notes, annotate or develop procedural flow diagrams, ask questions to clarify, and make it his/her objective to clearly understand the process. In particular, the auditor should be alert for indications of shortcuts or abbreviations to the procedure. During interviews, the auditor must remember that data are being gathered, and that conclusions should wait until all of the facts are in. This provides a clearer understanding of the actual processes used on the project and eases communications with the staff. The checklist developed during the preparation phase is used to guide the discussions during the interview.

Once the auditor is sure that the processes and procedures are understood as they really exist, he/she should begin examining the tangible parts of the project: its products and records. Products consist of requirements and design documentation, including unit development folders, user manuals, code, etc. Records consist of memoranda and forms that document the events in the life of a product. They come from CM, NRCA, and V&V, among others.

1. Records Examination

The auditor examines records to see if a procedure is being correctly followed. Record examination is described below in terms of the principal processes that SQA audits examine: CM, NRCA, and V&V. Similar activities would be used in the examination of other sets of records.

- ***CM Audit***

  During an audit of CM, the auditor should look at the complete change control cycle, beginning with the initial processing of a change request; through analysis of impact and dispositioning; design, code, and testing; updating of documentation; submission of the modified products to the library; and closure of the change request. Records to be examined include the change requests as processed by the Change Control Board, the work authorizing documents issued as a result of approved changes, the code and documentation products that are intended to reflect the approved changes, and the program library records that capture the changes to code and data. Throughout the audit, the auditor should be alert for and document any evidence of unauthorized changes.

  The records should show the authorization of each change, the product(s) to be changed, and the version numbers of the changed product. Much of the auditor's attention should be devoted to the Program Library or equivalent, since this is where the various versions of products and the change documents controlling those versions are stored. The auditor should check the products in the library to ensure that documentation is up-to-date with code changes. The auditor should check the version numbering and identification schemes, and the control documents. The records should demonstrate that there are adequate security measures in place to prevent loss and unauthorized changes. The auditor should verify that every item of code and documentation in the program library was properly received.

- ***NRCA Audit***

  When auditing the NRCA system, the auditor should look at the complete cycle. The auditor should review the nonconformance reports that are filed, to assure that they are completely and correctly filled out. The disposition process and board actions

should be recorded, usually on the same form. The nonconformances that result in product changes should be tracked to the product, and evidence should be gathered that changes are made, tested or reviewed, and approvals for issuance are granted. The NRCA procedures will parallel those used in CM, and can be audited in much the same way, especially when it comes to the program library. In both cases (CM and NRCA), the auditor should pay particular attention to corrected products to assure that they still satisfy requirements and standards.

- ***V&V Audit***

  An audit of V&V procedures should include a check of the verification matrix or equivalent, to assure that every requirement has a test and every test checks a requirement. Test plans should be adequate, specifying the test environment, test procedures, and the expected results for each test. Test procedures should be clear and detailed. Test plans and procedures should be reviewed and approved.

  The auditor should verify from SQA records that test procedures were followed and that all nonconformances observed during testing are recorded in the NRCA system. In addition to testing, the auditor should assess other methods of V&V, if used. For example, if inspections or another form of peer reviews are used to find problems, the auditor should verify that the records of the review show that they were done and that corrections and changes agreed to in the review are made in the product.

2. Product Examination

The intent of examination of products is two-fold: to see if standards are being followed, and to see if status is accurately reported. Documents are measured against documentation requirements to make sure that all required documents exist, and against documentation standards to ensure that they have the correct content and style. The auditor must read enough of the documents to form an opinion on the above; that is, the auditor must be able to determine that a document presented as showing the design indeed contains design information. On the other hand, the auditor is not responsible for the technical correctness of the documents and should not spend time trying to ascertain if the documents are correct.

Code also is examined to determine if it meets standards. Code standards are likely to include rules for internal documentation, size of modules, styling formats, and other such items that the auditor can verify. Rules for coding constructs or variable naming conventions are more difficult to verify. If the project has a code standards checker, the auditor may run it on some code. If the standards checker is to be run at a certain step in the development process, or if peer reviews are used to verify coding standards, the auditor must have access to those records.

Products also are examined to compare their status with that reported. Documents reported as complete, for example, should contain all of the sections given in the table of contents (which may be prescribed by a documentation standard), should be signed by the approving authorities, and should contain few, if any, To-Be-Determined (TBDs) items. Code implementation usually goes through the steps of detailed design, code, peer review, and unit test. A module that is reported as complete should have gone through all of the above steps, should meet the coding standards, and should have whatever approvals are required. The Unit Development Folder or equivalent should contain all of the evidence to look at status of coding.

3. Sampling

During the process of checking records and products, the auditor usually cannot examine each and every item; therefore, some sampling process must be used. The auditor must decide on sample sizes that can be accommodated in the site visit. The sample sizes must be balanced between completeness of coverage (some items from each product or set of records) and depth of coverage (number of items from a specific product or set of records). If the focus of the audit is limited, the sample size can be larger for the specific product or processes that are to be covered. In deciding on sample sizes, the auditor must allow time to follow up in more depth in areas where the initial sample indicates problems. The specific products or records to be included in the sample should be chosen by some "randomizing" method, and the project staff should not be informed in advance which items will be examined and which will not.

C. Audit Reporting

Once the interviews and record examination have been completed, initial results should be shared with the staff of the audited project during an exit interview. The exit interview provides an opportunity to clear up misunderstandings and allows project staff to present any information that they feel the auditor failed to consider. In addition, project staff learn immediately about the problems that have been found and can begin making plans to correct them.

After adjusting the initial results to reflect the information gathered in the exit interview, the auditor prepares a written final report. The report should be organized to highlight the most significant results, addressing both problems and commendations, and should include a general narrative of the audit. An example table of contents for an audit report is shown in Appendix A. The audit report should be addressed to the management official who arranged for the audit, if the audit is external; or directed as required by procedures, if internal.

The objective of the audit report is to present a clear picture of the status of a development activity or a facet of the activity to project management. The report must be clear, objective, and factual. In some cases, the auditor will find that, while procedures are being followed or standards are being met, the procedures or standards are not effective in producing a quality product. It is the responsibility of the auditor to note the specific problems caused by the procedure and/or standard and include them in the report. In general, however, problems that the auditor identifies should be related to project or contractually-required procedures and standards; the auditor's opinion of their desirability should not affect his/her evaluation of the adherence to them.

D. Follow-up

While the auditor's role is essentially finished after producing the audit report, actions to resolve deficiencies identified in that report must be taken by project management. Problems that are feasible and reasonable to correct should be converted to action items and assigned to appropriate individuals. A rationale should be developed for those that are not to be corrected. It is the responsibility of the developers to improve their processes in response to deficiencies identified by the audit. The changes should be tracked to ensure they occur and are effective and the closure of action items should be documented. In many cases, the best way to determine if the problems have been solved is through a follow-up audit.

IV. SQA AUDIT SCHEDULING

A. Routine Scheduling

Internal SQA audits should be scheduled frequently enough to identify potential problems so that no surprises develop for project management. They should be scheduled routinely during the life cycle, particularly around life cycle phase transitions. The most effective internal audit programs schedule frequent audits of small areas of project activity. Frequent auditing, combined with other SQA monitoring activities, would assure project management that the actual status of the project is known, vis-a-vis standards, procedures, and schedules.

External audits require more planning and interview time, but are scheduled much less frequently. The most important time for an external SQA audit is at the start of the implementation phase. This audit assures that the developer's standards and procedures are implemented in a manner appropriate for the project and that they are being followed. A second important time in a project's life cycle is the beginning of system integration. An external audit helps to assure that the software is ready for integration, that test plans and procedures are in place, and that procedures for control of the software are not short-circuited. Projects that are in trouble or have no internal audit function should have more frequent external audits.

Another factor to consider in the scheduling of audits, either internal or external, is the results of previous audits. Each SQA audit should include a review of the results and action items from any previous audits to confirm closure. If there were a number of problems and action items, audits should be scheduled more frequently. Projects that follow their procedures, meet their standards, and are accurate in reporting schedule and status need less frequent auditing.

B. SQA Audits in Response to Warning Signs

Some projects may show indications of problems in the development process. When warning signs appear, the acquirer should consider conducting an external audit as part of its response. The same warning signs can be used by the software provider to step up or evaluate the effectiveness of its internal audit program.

The audit program should be intensified if the project exhibits any of the following signs:

- Frequent schedule/milestone changes.
- Inconsistency of the developer's organizational structure with original plans or apparent inconsistency with the structure or functionality of the products to be produced.

- Unexplained fluctuation of project staff level or under- or over-staffing compared to estimates.

- Increases in the number of TBD items and action items without adequate progress in solutions.

- The inability or unwillingness of the developer to provide adequate and accurate information on project status, schedules, and plans.

- Continual delay of scheduled software system capabilities to later releases/versions.

- Unreasonable numbers of nonconformances or change requests; for example, a large number unresolved, or a sudden increase in numbers. An "unreasonable number" might be a suspiciously small amount of nonconformances for a complex system.

There may be other indications that are apparent to project management in specific cases. An experienced project manager's intuition that something may be wrong is a warning sign that should be heeded. An external audit is a cost effective way for an acquirer to ascertain the real product status and real processes being used by a developer; developer management should have an ongoing audit program to assure that no surprises are in store for them.

C. Announcing Audits

Adequate notification of audits should be provided to the developers for a number of reasons. Unannounced (surprise) audits are disruptive and demoralizing to the development staff and should be avoided. The intent of an audit program should be to help promote conformance with standards and procedures and the reporting of accurate status, not to "catch in the act" those "guilty" of violations. An announced schedule of audits allows proper preparation in terms of having required documentation available and being prepared to answer the auditor's questions.

V. SQA AUDITS DURING THE SOFTWARE LIFE CYCLE

A. Software Concept and Initiation Phase

During the concept and initiation phase, the software concept is developed, the feasibility of the software system is evaluated, the acquisition strategy is developed, and, if a contract is to be used to acquire the software, procurement is initiated and a contract is awarded. Before selecting an organization to perform a project, the acquiring organization can request a pre-award SQA audit. The intent of this type of audit is slightly different from audits performed later in the life cycle. Since there are no activities underway on the software that is to be developed, the auditor can only review the provider's "corporate" or generic standards and procedures, and past projects. If possible, these should be examined in the context of the proposed project, so that their effectiveness can be judged. This type of audit requires an experienced auditor.

The procedures and standards for the project are formulated during this phase. The SQA staff of the acquirer should ensure that standards and procedures adopted are appropriate for the project and are auditable, i.e., have a clear documentation trail, with easy-to-follow steps. They also should make sure that the contract allows external audits and requires internal audits.

B. Software Requirements Phase

During the software requirements phase, the software concept and allocated system requirements are analyzed and documented as software requirements. Test planning is begun, with a method for verifying each requirement identified and included in a preliminary test plan. Risks are identified and risk management control mechanisms are established. The size and scope of the remainder of the project is reevaluated, and changes in resources and schedules are made. Methods, standards, and procedures are detailed and put in place. The phase ends with a requirements review, at which the requirements are agreed to between the acquirer and developer and put under CM.

Internal audits during this phase concentrate on the process of developing, documenting, and controlling the requirements. Some process should be in place to control the requirements and draft documents as they are developed. This process probably will be relatively informal, and may include NRCA and an action item tracking system. There may be procedures for reporting on progress, estimating system and project resources, and risk assessment. All of these can be audited to the extent that controlled processes are in place. In addition to procedures, auditors should verify that requirements documents follow the format specified in the documentation standard.

An external audit, if one is performed during this phase, may look at the same items that are covered by an internal audit. In addition, an external audit can cover the same items as listed for a pre-award audit.

C. Software Architectural Design Phase

The objective of the architectural design phase is to develop an overall design for the software, allocating all of the requirements to software components. The software requirements are controlled and managed, and documents baselined following the requirements phase are changed only by a formal process. The phase ends with the preliminary design review, during which the acquirer and developer agree on the architecture of the system that is to be produced. Rework and action items resulting from the review are tracked and completed.

Internal and external audits during this phase should include the design documentation, verifying that format standards are met. The auditor should assure that all requirements are being allocated to software components. It is especially important to audit the configuration control mechanisms for the requirements to make sure that unauthorized and uncontrolled requirement change and growth is not occurring. In addition, items such as those mentioned in the previous phase, i.e., status reporting, action item tracking, and nonconformance reporting should be audited.

D. Software Detailed Design Phase

During the detailed design phase, the architectural design is expanded to the unit level. Interface control documents are completed and test plans revised. Constraints and object system resource limits are reestimated and analyzed, and staffing and test resources are validated. The phase ends with the critical design review, and the detailed design is baselined.

Audits during this phase should focus on the progress and documentation of the detailed design. If unit development folders (or other similar documentation) are used, they should be started during this phase, and can be audited. As auditing is done, reported status should be compared with the actual status. Any discrepancies should be noted. Both the requirements and the architectural design should be under CM and the CM process should be audited. Other items listed in the descriptions of the previous phases are still applicable.

E. Software Implementation Phase

During the implementation phase, the software is coded and unit tested. All documentation is produced in quasi-final form, including internal code documentation. At the end of the phase, all required products should be ready for delivery, subject to modification during integration and testing. Audits during this

phase check the results of design and coding, CM activities and program library, NRCA process, and schedule and status of the project.

Internal audits should be frequent during this phase. The project staff is usually at its maximum, and there are a great number of simultaneous activities. SQA auditing is one of the more important ways for management to keep the process under control, assure that quality products are being developed, and that status is actually as reported. Completed products are being sent to test as they are ready, and the products and their control process should be audited. Audits should include code audits to make sure coding standards are being followed and that internal code documentation standards are met. If inspections or some other form of peer reviews are done, the auditor should check that they are completed on all products and that action items resulting from them are carried out.

An external audit is most effective if done early in the implementation phase. At this point in the life cycle, all control procedures are in operation and all standards are in use. This external SQA audit assures that they are being followed correctly and that status is correctly reported. If any problems are noted, it is early enough for meaningful change and corrective action.

F. Software Integration and Test Phase

The objectives of the integration and test phase are to integrate the software units into a completed system, discover and correct any nonconformances, and demonstrate that the system meets its requirements. The phase ending review is the test readiness review, during which the developer provides to the acquirer evidence that the software system is ready for acceptance testing. During this phase, the test plan is executed, the documentation is updated and completed, and the products are finalized for delivery.

During this phase, internal audits include any and all of the items in previous phases. However, internal audits should concentrate on assuring that product changes made to correct nonconformances discovered during the testing are controlled, completed, and documented. Audits of the CM and NRCA processes, and computer program library are highly important. The SQA audit should include a check of the formal test procedures and the test results. Integration and test is often the most confusing and time-pressured part of a project, and there is a tendency to discard standards and procedures due to this pressure.

External audits during this phase should concentrate on the same items as internal audits, with additional emphasis on assuring completeness; that is, that testing has not been shortchanged in order to meet schedules.

G. Software Acceptance and Delivery Phase

During the acceptance and delivery phase, the formal acceptance procedure is carried out. As a minimum, there is a requirements-driven demonstration of the software to show that it meets those requirements. The process also may include acquirer tests, field usage, or other arrangements that are intended to assure that the software will function correctly in its intended environment.

This phase is very much like the end of the previous phase, with system tests being run, nonconformances noted, and corrections being made to the software, documentation, and data bases. The items to be audited are similar, especially the CM and NRCA processes.

H. Software Sustaining Engineering and Operations Phase

During this phase of the software life cycle, the software is used to achieve the objectives for which it was acquired. Corrections and modifications are made to the software to sustain its operational capabilities and to upgrade its capacity to support its users. Software changes may range in scope from simple corrective action up to major modifications that require a full life cycle process.

Internal audits should respond to the extent and type of changes being made to the system. If there is only a low level of corrective action, then audits may be limited to the CM and NRCA procedures and to verifying that quality is being maintained in the products. If substantial modifications are being made, however, then a full or mini-life cycle should be in place and audits should be performed as described for the appropriate stage.

When long term sustaining engineering is being performed, an external audit should be done periodically to assure the acquirer that product quality is maintained and sustained. A minimum of one external audit per year is recommended; more if the level of change activity is high.

VI. PREPARING A CHECKLIST

An audit checklist is a list of items that the auditor intends to examine and questions the auditor intends to ask during the site visit portion of the audit. While a generic checklist may be used as a basis for all audits, better results will be achieved if the generic checklist is tailored for each audit. Tailoring consists of choosing appropriate items or questions from the generic checklist, expanding the level of detail, adding additional questions and topics, and changing the wording of the questions to fit the project's nomenclature. Information for tailoring may come from the contract requirements, organizational standards and practices, and results and action items from previous audits. Additional information to be considered for tailoring should include the structure of the development organization and project, life cycle phase, and audit focus.

In developing the checklist, the auditor should be careful not to overlook important information that appears to be obvious. For example, assuming the project has a product specification may be erroneous; adding that item to the checklist will help to assure that the information is confirmed.

A sample generic checklist, divided by topic, is provided in Appendix B. Under each topic is a series of typical questions that should be addressed if that topic is going to be part of the audit. To tailor this checklist, the auditor should determine which topics apply to the audit and whether questions should be answered by interviews, examination of the software products and documents, examination of records, or a combination of methods. The auditor then should sort the questions by the method that is intended to be used to answer them, and further, by the precise source to be used. For example, questions about how CM operates might be asked of the CM manager during an interview, but some of those same questions might be directed at the person who operates the project's computer program library. Answers to other CM questions might be found through an examination of the records of the CM process; still others by an examination of code and documentation products.

As much as possible, questions should be phrased in terms of the specific project and organization being audited, and should use the names and terms that the project uses. This tailoring will take some work on the part of the auditor, but this effort will be repaid by the fact that effective communication will be established earlier.

The parts of the tailored checklist that will be answered by an examination of records or products should be put on a form for use on-site. The form can be simple, but should allow space for answers to each question and additional comments. The form should, if possible, allow the checking of boxes or simple entry of information. A sample form for the recording of a CM audit is shown in Appendix C.

As the auditor proceeds with the site visit, the checklists and forms can be completed with the information obtained. The auditor must retain the flexibility to modify the forms or questions as information is gathered. Additional questions are likely to be suggested by answers given, and forms may not have been properly made in advance to record the real situation. It is important to remember that the checklist and forms derived from it are guides, and that the objective of the audit is to understand and report on the actual state of affairs in the developing organization.

VII. AUDITING IN THE ABSENCE OF STANDARDS AND PROCEDURES

An auditor may be asked to "audit" a project that lacks documented standards and procedures, perhaps because of warning signs indicated in Chapter IV. Most often, this type of audit will be external to the project, even if the auditor is employed by the developing organization, because a developer that does not have documented standards and procedures is unlikely to have an internal audit program.

All projects generate code and documentation, but if there are no written standards, the products will be in the style of the individual technical performers or their managers. All projects handle changes and problems, and test their software. The methods may be somewhat ad-hoc and dependent on the specific individuals involved in a specific case, but they do exist, documented or not. The role of the auditor is to discover and document the "standards" and "procedures" that are actually followed.

After the auditor has determined from interviews what "standards" and "procedures" are followed, the rest of the audit can proceed like any other audit. That is, the auditor can follow the progress of control paths and determine the extent to which the procedures are followed versus the number of exceptions that are allowed. The auditor can sample the products and rate their conformance to the (unwritten) standards.

The auditor must gather enough information to evaluate the suitability and consistency of the unwritten standards and procedures. The auditor may be experienced enough to do the evaluation, or the auditor may wish to leave the evaluation to the management to which he/she will report. In either case, the auditor has to gather information on product quality, consistency of application of the unwritten rules, the adequacy of testing and reviews, and instances of confusion and/or error that may have resulted from uncertainty. This information is then used for evaluation.

VIII. QUALITIES OF AN AUDITOR

The major contribution of an internal or external auditor to project success is the collection and presentation of information that allows project management a clear view of the product's actual status and the actual compliance with standards and procedures. This requires an impartial auditor. In particular, an internal auditor must remember that covering up problems, due to feelings of empathy with the project staff or a desire to present the developer's organization in a good light, is counterproductive. Problems that are not brought to light will not be solved, and may result in much larger problems later in the life cycle.

A good auditor should have a basic understanding of the software development life cycle and the products and processes involved in each of its phases. If an auditor is expected to evaluate the standards and procedures used by the developer and to judge their impact on product quality and project schedule, then he/she needs significant experience and background in software development and software management. It helps if the auditor is knowledgeable about the type of software being audited, and is aware of the specific software development procedures used in the project. It is useful if the auditor is experienced or trained in auditing techniques.

IX. TECHNIQUES AND TOOLS

The most frequently used tool for an SQA audit is an audit checklist. The checklist must be tailored to the project to be audited, as it provides a list of questions that must be answered about that particular project.

Automated tools, either brought by the auditor or provided by the project, may be used if compatible with the project's standards and procedures. For example, the project may have a standards checker for code. The auditor can run the checker on a sample of the code, or can verify that the project runs the checker.

The checklist tailoring and form-making process also may be assisted by keeping a generic checklist in a database or word processor. The tailored information may then be automatically transferred to a form or brought to the audit on a laptop computer.

APPENDIX A

SQA AUDIT REPORT

The following is the minimum content for an SQA audit report.

1. Background
   a. Identity of audit
   b. Date of audit
   c. Audit team members
   d. Current phase of development

2. Findings
   a. Version of products audited
   b. Anomalous conditions encountered
   c. Recommendation for each anomalous condition (if applicable)

3. Summary
   a. Summary of findings
   b. Status
   c. Date of follow-up or next scheduled audit

APPENDIX B

SQA AUDIT CHECKLIST QUESTIONS

The following is a sample master list of questions that can be tailored for an SQA audit. Questions appropriate to a specific audit should be selected and then modified to reflect local terminology or procedures. The questions should be placed on a form that allows space for recording answers.

Questions shown in *italics* are mainly for use in the staff interviews.

Software Assurance

Has an SQA plan been prepared? Is it maintained current with program requirements?

Has the SQA plan been submitted for approval?

Does the SQA plan include or define:

- SQA requirements and activities to be implemented?
- Schedule showing when each of the activities will be implemented?
- Budget for activities?
- Specific organizational assignments?
- Interaction between SQA and the overall development effort?
- SQA participation in the overall change management process?
- SQA participation in the overall test process?

Is there evidence that SQA planned activities are implemented throughout the life cycle?

Development Documentation

Are standards for preparation of deliverable documentation established?

Does the documentation meet the standards?

Are procedures established and documented to assure that standards are followed?

Do the procedures address the changes to software documentation that are placed under configuration management control? *Are the changes reviewed in the same manner as the base document?*

Are methods established for traceability of documentation, including changes?

Are the contents of deliverable documents clear, concise, complete, and understandable?

Are procedures established to enforce consistency in writing?

Are review teams familiar with the material being reviewed to detect inconsistency?

Is approval authority for deliverable documentation clearly stated?

Is required documentation provided to the acquirer in a timely, responsive manner?

Are sufficient copies furnished?

Are established procedures followed in the production of both deliverable and nondeliverable documents?

Does the documentation in the development folder match the phase of the life cycle?

Does the level of detail in documentation look reasonable?

Code

Do code, prolog, and Program Design Language (PDL) adhere to all prevailing standards and conventions?

Are necessary elements of the prolog complete; e.g., are all data elements described, all subroutines defined?

Is internal code documentation present in amounts required by standards?

Is the code consistent with its design, as presented in its prolog and PDL?

Does the code appear to be correct for test cases that can be verified by a quick, visual inspection?

Is all debug code clearly identified?

Are all stubs and test files identified?

Do test cases appear adequate based on the PDL?

Configuration Management

Has a software configuration management (CM) plan been developed? Has the plan been baselined? *Provided to the acquirer?*

Are CM instructions for identification of baseline items and subsequent revision or versions being followed?

Are CM procedures in place which require approval authority for adding and removing items in the program library?

Is the CM organization adequately staffed, fully funded, and responsive? Are responsibilities clearly understood?

Do baseline documents comply with contract requirements?

Do the approved specifications serve as a baseline for control of changes?

Is a list of approved specifications maintained? Current? Changes posted?

Are procedures established for the production of software documentation adequate and rigidly enforced?

Are procedures for handling problem reports adequate and efficient?

Has a Configuration Control Board (CCB) been established? Who are the members? Is SQA represented? *Do all members attend regularly? Are CCB actions handled in a timely manner?* Are agenda and minutes published? *Are CCB action items followed up?*

Are CM status accounting documents maintained? Are they current?

Does the CM plan address configuration audits?

Have formal configuration audits been conducted or planned (including FCA and PCA)?

Computer Program Library

Has a Computer Program Library been established? A program librarian appointed?

Have adequate procedures been identified for: Library controls? Configuration item controls? Problem report handling?

Is the program librarian complying with established procedures?

Are problem reports implemented into appropriate development folders?

Are computer program versions accurately identified, controlled, and documented through the life cycle? Is an automated source control system used? *Is it adequately maintained?*

How is the library controlled (error report, change request, etc.)?

Are only authorized/approved modifications made to source and object programs released to the library? How is it controlled (error report, change request, etc.)?

What measures are being taken to assure all approved modifications are properly integrated and that software submitted for testing is the correct version?

Is nondeliverable software monitored and controlled to the extent specified in the development plan?

Are development folders regularly submitted to the program librarian?

Does a library documentation index exist? Is it current?

Does a log exist showing what material has been checked in and out of the library? Does it appear accurate?

Does all submitted code include proper transmittal information? Is this available for review?

Is documentation updated to correspond with newly submitted code?

Are all items placed in the program library assigned an identification number related to the version number? Does this number relate to the associated documentation?

Is the flow through a change cycle clear, efficient, documented, and correct? (Test several samples.)

Nonconformance Reporting and Corrective Action

Have procedures assuring prompt detection and correction of deficiencies been established?

Are data analyzed and problem and deficiency reports examined to determine extent and causes?

Are trends in performance of work analyzed to prevent development of nonconforming products?

Has corrective action been documented accurately on problem reports?

Has corrective action been reviewed and monitored to determine adequacy, effectiveness, and whether contract requirements are being met?

Are all corrective action reports and analyses on file?

Is there management support for the corrective action system?

Is the program librarian following procedures for maintaining control and status of problem reports?

Are discrepancies generated by nondeliverable computer programs treated the same as those for deliverables?

Are problem reports pertaining to a unit contained within the development folder for that unit?

Are the software developers complying with the requirement to generate problem reports during integration?

Is there documented approval for all changes to items under configuration control? Do all forms have required signatures?

Verification and Validation

Have the software requirements been analyzed to determine testability?

Are test objectives adequate, feasible, and sufficient to demonstrate software performance to meet contractual requirements? *Are they understood by project personnel?*

Are the test philosophy and methodology based on assumptions that are acceptable to SQA? Is there a procedure to monitor assumptions and a way to alert the test director if an assumption is unacceptable?

Do test plans and procedures comply with specified standards and contractual requirements?

Are the test plans and procedures approved by the acquirer, where required?

Are all test tools and equipment identified, defined, calibrated, and controlled prior to testing the software? Is all necessary test hardware certified (both computer and ancillary)?

Is software baselined prior to testing?

Are the correct version of software and associated documentation certified prior to testing?

Are acceptance tests monitored by an SQA representative? By the acquirer, when required? *If not, then who monitored the tests?*

Are tests conducted according to test plans and procedures?

Have test results been certified by participating members to reflect the actual test findings?

Have test reports been reviewed and certified? *By whom?* Are deficiencies documented in problem reports?

Has test-related documentation been maintained and controlled to allow repeatability of tests?

Is there a test verification matrix to assure all requirements are tested? Does it look reasonable?

Are test procedures clear and repeatable?

Do actual and expected test results match? If not, has a problem report been filed?

Project Status

Do completion dates in development folders/status sheets agree with status report to management? If not, how great is the difference?

According to the development/management plan, the project where it should be? *What activities should be current? How should the project be staffed? What intermediate projects should be delivered? What reviews or milestones should have occurred?*

Where does the project actually stand now? Determine:

- *Current phase*
- *Activities levels*
- *Staff composition*
- *Documents delivered*
- *Milestones reached*
- *Results of reviews.*

APPENDIX C

SQA AUDIT CHECKLIST

Project:

Auditor:

Audit Date::

Question	YES	NO	N/A	Remarks
Has a Computer Program Library been established?				
A program librarian appointed?				
Have adequate procedures been identified for: Library controls?				
Configuration item controls?				
Problem report handling?				
Is the program librarian complying with established procedures?				
Are problem reports implemented into appropriate development folders?				
Are computer program versions accurately identified, controlled, and documented through the life-cycle?				
Is an automated source control system used?				

INDEX

SOFTWARE QUALITY ASSURANCE AUDITS GUIDEBOOK

CHANGE SUGGESTION FORM

Name:_____ Center/Contractor:_____ Phone:_____

Version:_____ Chapter:_____ Section:_____ Page:_____

Suggested Change:

Rationale:

Forward to Donald W. Sova
Manager, SMAP
NASA Headquarters, Code QR
Washington, DC 20546